Posture Optimization in Robotic Flat-End Milling Based on Sequential Quadratic Programming

Yongxue Chen
State Key Laboratory of Mechanical System and Vibration;
School of Mechanical Engineering,
Shanghai Jiao Tong University,
Shanghai 200240, China
e-mail: cyxsjtu@sjtu.edu.cn

Ye Ding[1]
State Key Laboratory of Mechanical System and Vibration;
School of Mechanical Engineering,
Shanghai Jiao Tong University,
Shanghai 200240, China
e-mail: y.ding@sjtu.edu.cn

Robotic flat-end milling of complex surfaces offers advantages such as high flexibility and high machining efficiency. In the process of planning the toolpath based on the cutter contact path, the robot functional redundancy and the tool orientation need to be solved carefully. This paper presents a posture optimization method for robotic flat-end milling. Taking the weighted sum of the machining width and the toolpath smoothness performance criterion as the objective function, an optimization model considering the joint limits and gouging avoidance is established. An efficient algorithm based on sequential quadratic programming is proposed to solve this nonconvex problem. During the execution of the algorithm, the machining width is efficiently calculated by an iterative method based on conformal geometric algebra, while its derivatives are approximated analytically. Simulations and experiments demonstrate that the presented technique can resolve the tool axis direction and the robot redundancy effectively to increase the machining width and improve the toolpath smoothness, thus reducing the time for machining and improving the surface quality. [DOI: 10.1115/1.4056707]

Keywords: robotic milling, toolpath optimization, sculptured surface, flat-end cutter, sequential quadratic programming, CAD/CAM/CAE, robotics and flexible tooling

1 Introduction

Due to some outstanding features, such as a large workspace and excellent flexibility, industrial robots (IRs) for machining have gained extensive attention. However, compared to five-axis computer numerical control (CNC) machine tools, the low rigidity, low following accuracy, and complex kinematics of IRs hinder their applications.

When IRs are used for five-axis machining tasks, the degree-of-freedom (DoF) of the tool rotation around the tool axis is functionally redundant. Many methods to improve machining accuracy and efficiency by optimizing this redundant DoF have been developed in the existing literature. To improve the stiffness of the IRs, Xiong et al. [1] presented a frame-invariant stiffness performance index, and a one-dimensional discretization search algorithm was used to optimize the redundancy of each cutter location (CL). Liao et al. [2] presented a method to resolve the robot redundancy and the workpiece setup at the same time; thus, the stiffness was further improved. Based on the deformation induced by cutting force and the weight of the spindle, Chen et al. [3] introduced a composite deformation index to assess the stiffness. Léger and Angeles [4] minimized the condition number of the Jacobian matrix by using sequential quadratic programming (SQP). Since the goal of increasing the stiffness is to improve machining accuracy, Lin et al. [5] took the contour error as the direct optimization objective. On the other hand, the smoothness of robot joint paths has a significant impact on the machining quality, and many works focus on this criterion. Peng et al. [6] presented a smoothness criterion based on joint acceleration and used the sequential linear programming method to resolve the redundancy globally. On the basis of the differential evolution (DE) algorithm, Lu et al. [7] proposed an optimization method that achieved the smoothness of joint paths and collision-free manipulation.

All of the above methods take the redundant DoF as an optimization variable. However, the tool axis direction also affects the joint paths of the robot, so taking the three DoFs of the tool posture as optimization variables is a more reasonable choice. Based on it, some posture optimization methods for robotic ball-end milling considering the constraints of tool axis direction have been proposed**.** For example, Lu et al. [8] introduced a sampling-based method to minimize the arc length of the joint paths. By optimizing the tool pose differential vector, the joint-smooth toolpath was obtained in Ref. [9]. Liao et al. [10] improved the stiffness by applying a region-based method. Li et al. [11] used a virtual repulsive potential field method to optimize the tool posture considering mechanical and kinematic constraints. However, to the best of our knowledge, few works focus on posture optimization in robotic flat-end milling.

Compared to ball-end milling, flat-end milling is more desirable in many cases of surface machining since it has higher machining accuracy and efficiency [12,13]. However, the orientation planning of the flat-end milling is more complicated, since the tooltip point needs to change when the tool axis direction is changing to keep the cutter contact (CC) point constant. To improve the machining width while preventing gouging with the design surface, the problem of orientation optimization for five-axis flat-end milling has been extensively studied. Chiou and Li [14] determined the optimal tool orientation by using a swept envelope approach, and they found that the curvature matching between the tool envelope surface and the local design surface helps increase the machining width. Fard and Feng [13] presented the influence of tool tilt angle and inclination angle on the machining width, and they proposed an efficient approach to resolve the feed direction and the tool orientation by using a toroidal surface to approximate the local surface [15]. Gong et al. [16] derived a second-order approximation of the cutter-swept envelope, and a tool orientation adjusting method at each CC point was proposed based on the approximation method [17]. By using two quadric surfaces to approximate the surface profile of the workpiece, Fan and Ball [18] developed an orientation planning approach to improve the

[1]Corresponding author.

Manuscript received June 30, 2022; final manuscript received January 16, 2023; published online February 10, 2023. Assoc. Editor: Qiang Huang.

machining efficiency. Taking both the machining width and the toolpath smoothness into consideration, Lu et al. [19] resolved the feed direction and the tool orientation at each CC point based on the DE algorithm. Later, a global optimization method was proposed by them [20]. In this method, all variables along the entire CC path are optimized simultaneously, and the efficiency of machining the whole sculptured surface is improved further.

As mentioned earlier, simultaneous optimization of robot functional redundancy and tool orientation in robotic flat-end milling to improve machining width and path smoothness is a challenge. To solve this problem, a new SQP-based optimization method for the entire CC path is presented in this paper. Firstly, an optimization model considering the joint limit and gouging avoidance is established. In this model, the objective is taken as the weighted sum of the machining width and the proposed smoothness performance criterion of the joint paths. Then, an efficient method based on SQP is proposed to solve this nonconvex problem. To further decrease the algorithm running time, an efficient iterative method for calculating the machining width, as well as an analytical approximation of its derivatives, is proposed with the help of conformal geometric algebra (CGA). Finally, multiple simulations and experiments are conducted to verify the validity and efficiency of the proposed approach.

The remainder of the paper is organized as follows. In Sec. 2, the background for robotic flat-end milling is briefly reviewed. In Sec. 3, the optimization model and the SQP-based algorithm are presented. Section 4 presents the efficient methods for calculating the machining width and its derivatives. In Sec. 5, simulations and experiments are presented to verify the validity of the developed method. Section 6 gives conclusions.

2 Background for Robotic Flat-End Milling

2.1 Kinematic Model of the Robotic Flat-End Milling System. For the robotic flat-end milling system shown in Fig. 1, the CC path is defined by a set of CC points $\{\boldsymbol{CC}_i | i = 1, 2, \ldots, n\}$ on the design surface, which is represented in the workpiece coordinate system $\{WCS\}$. $\{LCS_i\}$ shown in Fig. 2 is a local coordinate system constructed at CC point $\boldsymbol{CC}_i$. The x-axis of $\{LCS_i\}$ is the same as the feed direction, and the z-axis is set to the normal vector of the design surface. The y-axis of the system is obtained by the right-handed rule. On this basis, the orientation of the flat-end tool is commonly determined by two rotational angles. The tool is rotated around the y-axis of $\{LCS_i\}$ for an inclination angle λ_i at first and then around the z-axis for a tilt angle ω_i, as presented in Fig. 2. A tool coordinate frame $\{TCS0\}$ is established to represent the tool orientation, and its origin coincides with the center of the bottom tool circle (BTC). The z-axis of $\{TCS0\}$ coincides with the tool axis, and its x-axis passes through the CC point.

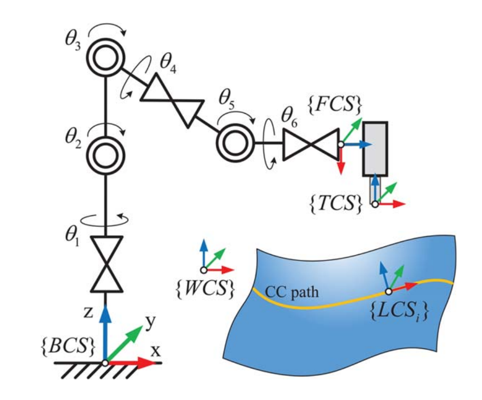


Fig. 1 Schematic of the robotic milling system

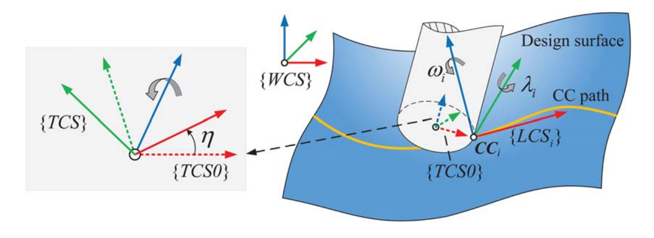


Fig. 2 Local, tool, and work coordinate system

The transformation between $\{LCS_i\}$ and $\{TCS0\}$ can be described by

$$\boldsymbol{T}_{TCS0}^{LCS_i}(\lambda_i, \omega_i) = \boldsymbol{R}_z(\omega_i)\boldsymbol{R}_y(\lambda_i)\boldsymbol{T}_x(-r) \tag{1}$$

where r denotes the radius of BTC, and $\boldsymbol{R}_z$, $\boldsymbol{R}_y$, and $\boldsymbol{T}_x$ are three homogeneous transformation matrices with the following expressions:

$$\boldsymbol{R}_z(\omega_i) = \begin{pmatrix} \cos(\omega_i) & -\sin(\omega_i) & 0 & 0 \\ \sin(\omega_i) & \cos(\omega_i) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2}$$

$$\boldsymbol{R}_y(\lambda_i) = \begin{pmatrix} \cos(\lambda_i) & 0 & \sin(\lambda_i) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\lambda_i) & 0 & \cos(\lambda_i) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3}$$

$$\boldsymbol{T}_x(-r) = \begin{pmatrix} 1 & 0 & 0 & -r \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4}$$

when IRs perform milling tasks, the functionally redundant DoF is introduced by rotating the end-effector around the tool axis. To determine the robot configuration, a tool coordinate system $\{TCS\}$ is constructed to represent the pose of the end-effector. $\{TCS\}$ is obtained by rotating $\{TCS0\}$ around its z-axis by an angle η_i, and this angle is defined as the redundant rotation angle, as shown in Fig. 2. On this basis, the transformation between $\{LCS_i\}$ and $\{TCS\}$ is

$$\boldsymbol{T}_{TCS}^{LCS_i}(\lambda_i, \omega_i, \eta_i) = \boldsymbol{T}_{TCS0}^{LCS_i}(\lambda_i, \omega_i)\boldsymbol{R}_z(\eta_i) \tag{5}$$

Note that the three angles $\{\lambda_i, \omega_i, \eta_i\}$ determine the tool posture at CC point $\boldsymbol{CC}_i$ in robotic flat-end milling.

Equation (6) is the kinematic model of the robot system illustrated in Fig. 1, from the robot base coordinate system $\{BCS\}$ to $\{TCS\}$

$$\boldsymbol{T}_{WCS}^{BCS}\boldsymbol{T}_{LCS_i}^{WCS}\boldsymbol{T}_{TCS}^{LCS_i}(\lambda_i, \omega_i, \eta_i) = \boldsymbol{T}_{FCS}^{BCS}(\boldsymbol{q}_i)\boldsymbol{T}_{TCS}^{FCS} \tag{6}$$

where $\boldsymbol{T}_{FCS}^{BCS}$ denotes the transformation describing the position and posture of robot flange coordinate system $\{FCS\}$ with respect to (w.r.t.) $\{BCS\}$, $\boldsymbol{q}_i = [\theta_1^i, \theta_2^i, \theta_3^i, \theta_4^i, \theta_5^i, \theta_6^i]^T$ is the joint angle vector, θ_j^i denotes the jth robot joint angle at the ith CC point, and $\boldsymbol{T}_{TCS}^{FCS}$, $\boldsymbol{T}_{WCS}^{BCS}$, and $\boldsymbol{T}_{LCS_i}^{WCS}$ are three constant homogeneous transformations describing the position and posture of $\{TCS\}$ w.r.t. $\{FCS\}$, $\{WCS\}$ w.r.t. $\{BCS\}$, and $\{LCS_i\}$ w.r.t. $\{WCS\}$, respectively.

2.2 Machining Width. The uncut part between adjacent toolpaths is called a scallop. For a preset scallop height h, i.e., the maximum acceptable undercut, the offset surface after shifting the design surface by a distance h is denoted as the scalloped surface. Denote the intersection of BTC and the scallop surface as $\boldsymbol{p}_a$ and $\boldsymbol{p}_b$, as illustrated in Fig. 3. Define the machining width at $\boldsymbol{CC}_i$ as [13]

$$w_i = |(\boldsymbol{p}_a - \boldsymbol{p}_b) \cdot \boldsymbol{y}_{LCS_i}| \tag{7}$$

where $\cdot$ denotes the dot product, $|\cdot|$ represents the absolute value, and $\boldsymbol{y}_{LSC_i}$ is the unit vector in the y-axis direction of $\{LCS_i\}$.

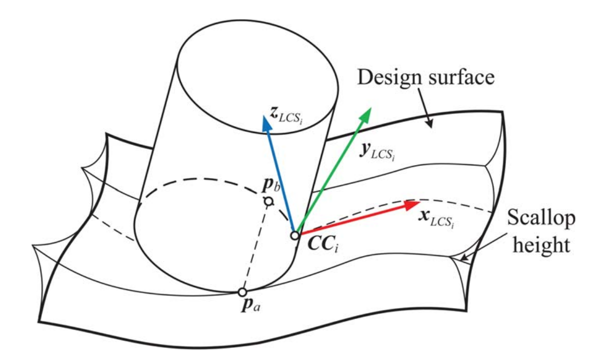


Fig. 3 Cut geometry in flat-end milling

2.3 Tool Gouging. Generally speaking, there are three kinds of gouges [21]: local, rear, and global gouges, as shown in Fig. 4. The main cause of local gouge is the curvature misfit at the CC point between the design surface and the tool envelope surface. If the part of the BTC that interferes with the design surface is not in the vicinity of the CC point, the interference is referred to as a rear gouge. The global gouge happens when the tool's non-cutting part collides with the workpiece. Since global gouge occurs much less frequently than the other two types of gouges in sculptured surface machining, only local and rear gouges are considered in this work.

Many methods were proposed to determine the gouging-free tool orientation, such as the multi-point machining method [22,23], rolling ball method [24,25], arc-intersect method [26], intersection judging method [27,28], and tool swept surface-based method [29,30]. At $\boldsymbol{CC}_i$, for any tilt angle $\omega_i \in [-\pi/2, \pi/2]$, the lower bound of λ_i can be obtained effectively and efficiently by using the above-mentioned methods. Define the lower bound of λ_i corresponding to ω_i at $\boldsymbol{CC}_i$ as $f_i(\omega_i)$, where f_i is a function of ω_i and can be derived by interpolating some sample points. To avoid tool gouging, the following conditions need to be met:

$$\begin{cases} f_i(\omega_i) \le \lambda_i \le \pi/2 \\ \omega_i^{low} \le \omega_i \le \omega_i^{high} \end{cases} \tag{8}$$

where ω_i^{low} and ω_i^{high} are the allowable minimum and maximum values of ω_i respectively.

Note that the above gouging-free conditions can be obtained in a relatively efficient manner for any CC point and that its simplicity of form has a positive impact on the efficient execution of the posture optimization algorithm in Sec. 3.

3 Toolpath Optimization Model and Algorithm

3.1 Smooth Criterion for Toolpaths. The improvement of the joint path smoothness is conducive to processing productivity and surface quality. The smoothness index of the five-axis machining proposed in Ref. [20] is extended to measure the robot path smoothness in this section, and it is defined as

$$\Phi_{smooth} = k_1 \int_0^l \sum_{j=1}^{6} \left(\frac{d\theta_j}{ds}\right)^2 ds + k_2 \int_0^l \sum_{j=1}^{6} \left(\frac{d^2\theta_j}{ds^2}\right)^2 ds \tag{9}$$

where s is the arc length parameter of the CC path, l is the length of the CC path, θ_j is the jth joint variable of the robot, and k_1 and k_2 are the weight coefficients. A characteristic of this index is that it is completely defined by the path geometry and is independent of the feedrate. If the tool contact point moves at a fixed speed on the CC path during machining, the index represents the integration of the weighted square sum of the joint velocities and accelerations along the CC path.

In practice, for the values of k_1 and k_2, a good choice is to make the values of the two terms before and after the plus sign in Eq. (9) essentially the same in order of magnitude. If more importance is given to a certain term, the corresponding weight can be adjusted as needed.

Denote the first-order derivative and the second-order derivative of the jth joint variable at $\boldsymbol{CC}_i$ w.r.t. the arc length parameter of the CC path as ϕ_1^{ij} and ϕ_2^{ij}, respectively. According to the numerical difference method for unequally spaced discrete points [31], for $i = 2, 3, \ldots, n-1$, ϕ_1^{ij} and ϕ_2^{ij} are approximated as

$$\phi_1^{ij} = \frac{\Delta s_{i-1}^2 \theta_j^{i+1} + (\Delta s_i^2 - \Delta s_{i-1}^2)\theta_j^i - \Delta s_i^2 \theta_j^{i-1}}{(\Delta s_i + \Delta s_{i-1})\Delta s_{i-1} \Delta s_i} \tag{10}$$

$$\phi_2^{ij} = \frac{2\theta_j^{i-1}}{\Delta s_{i-1}(\Delta s_i + \Delta s_{i-1})} - \frac{2\theta_j^i}{\Delta s_{i-1}\Delta s_i} + \frac{2\theta_j^{i+1}}{\Delta s_i(\Delta s_i + \Delta s_{i-1})} \tag{11}$$

where $\Delta s_i = \|\boldsymbol{CC}_{i+1} - \boldsymbol{CC}_i\|$, and $\|\cdot\|$ represents the two-norm of the vector. For the cases of $i = 1$ and $i = n$, ϕ_1^{ij} and ϕ_2^{ij} are approximated as

$$\phi_1^{1j} = \frac{\theta_j^2 - \theta_j^1}{\Delta s_1} \tag{12}$$

$$\phi_1^{nj} = \frac{\theta_j^n - \theta_j^{n-1}}{\Delta s_{n-1}} \tag{13}$$

$$\phi_2^{1j} = \frac{\theta_j^3 - \theta_j^2}{\Delta s_1 \Delta s_2} + \frac{\theta_j^1 - \theta_j^2}{\Delta s_1^2} \tag{14}$$

$$\phi_2^{nj} = \frac{\theta_j^{n-2} - \theta_j^{n-1}}{\Delta s_{n-1}\Delta s_{n-2}} + \frac{\theta_j^n - \theta_j^{n-1}}{\Delta s_{n-1}^2} \tag{15}$$

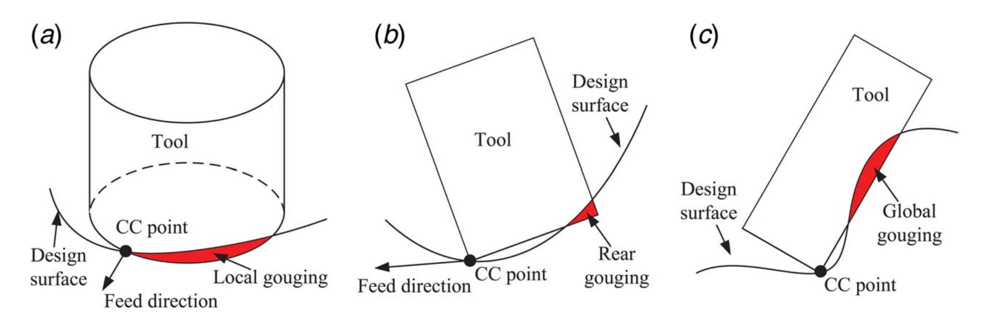


Fig. 4 Three types of gouging in flat-end milling: (a) local gouging, (b) rear gouging, and (c) global gouging

Then, the discrete form of Eq. (9) can be derived as

$$\Phi_{smooth} = \sum_{i=1}^{n}\sum_{j=1}^{6}[k_1(\phi_1^{ij})^2 + k_2(\phi_2^{ij})^2]ds_i \tag{16}$$

where

$$ds_i = \begin{cases} \Delta s_1/2,\ i=1 \\ \Delta s_{n-1}/2,\ i=n \\ (\Delta s_{i-1}+\Delta s_i)/2,\ i=2,\ \cdots,\ n-1 \end{cases} \tag{17}$$

3.2 Optimization Model. When machining a sculptured surface, a larger machining width results in a smaller number of toolpaths, thus improving the machining efficiency. For a CC path, the only critical value among all the machining widths is the minimum one, so it should be maximized. Considering the smoothness criterion shown in Eq. (16), the toolpath optimization problem has multiple objectives. In addition, during the machining process, the interference-free condition shown in Eq. (8) should be met, and the movement of the joints should be limited. Let $\boldsymbol{\alpha} = [\lambda_1, \omega_1, \eta_1, \cdots, \lambda_n, \omega_n, \eta_n]$, the toolpath optimization problem has the following form:

$$\textbf{P1}\quad \begin{array}{ll} \min\limits_{\boldsymbol{\alpha}} & \Phi_{smooth} - k_3 \min\limits_{i}[w_i(\lambda_i, \omega_i)] \\ \text{s.t.} & \boldsymbol{q}_i = f^{-1}(\lambda_i, \omega_i, \eta_i, \boldsymbol{CC}_i) \\ & \boldsymbol{q}_{\min} \le \boldsymbol{q}_i \le \boldsymbol{q}_{\max} \\ & f_i(\omega_i) \le \lambda_i \le \pi/2 \\ & \omega_i^{low} \le \omega_i \le \omega_i^{high} \\ & i = 1, 2, \cdots, n \end{array}$$

where $f^{-1}(\cdots)$ represents the robot inverse kinematics with closed-form, $\boldsymbol{q}_{\min}$ and $\boldsymbol{q}_{\max}$ are the allowable minimum and maximum values for the joint variable vector, k_3 is the weight of machining width. It should be noticed that setting the objective function as the weighted sum of multiple criteria is a commonly used mathematical technique for dealing with multi-objective optimization problems. Weight k_3 could be chosen as needed, and it should be ensured that the values of two terms in the objective function are comparable on the same level based on the initial solution.

By introducing a new optimization variable ψ, **P1** can be reformulated as

$$\textbf{P2}\quad \begin{array}{ll} \min\limits_{\boldsymbol{\alpha},\psi} & \Phi_{smooth} + k_3\psi \\ \text{s.t.} & \psi \ge -w_i \\ & \boldsymbol{q}_i = f^{-1}(\lambda_i, \omega_i, \eta_i, \boldsymbol{CC}_i) \\ & \boldsymbol{q}_{\min} \le \boldsymbol{q}_i \le \boldsymbol{q}_{\max} \\ & f_i(\omega_i) \le \lambda_i \le \pi/2 \\ & \omega_i^{low} \le \omega_i \le \omega_i^{high} \\ & i = 1, 2, \ldots, n \end{array}$$

3.3 Optimization Algorithm. Problem **P2** is a nonconvex optimization problem. Owing to the relatively large size of the problem and the extremely nonlinear relationship between the machining width and the optimization variables, the efficiency and effectiveness of solving this problem by using most general-purpose numerical optimization algorithms are unacceptable. In this section, an algorithm based on SQP is proposed to take advantage of the special form of Problem **P2** and solves it by sequentially solving a cascade of sparse quadratic programming (QP) problems. A distinctive feature of the algorithm is that there is no need to compute the Hessian matrix of the objective function or its approximation, which is often required by traditional SQP algorithms.

According to the robot inverse kinematics, the first-order differential increment of $\boldsymbol{q}_i$ is obtained by

$$\Delta\boldsymbol{q}_i = \left[\frac{\partial\boldsymbol{q}_i}{\partial\lambda_i}, \frac{\partial\boldsymbol{q}_i}{\partial\omega_i}, \frac{\partial\boldsymbol{q}_i}{\partial\eta_i}\right][\Delta\lambda_i, \Delta\omega_i, \Delta\eta_i]^T \tag{18}$$

where $\partial\boldsymbol{q}_i/\partial\lambda_i$, $\partial\boldsymbol{q}_i/\partial\omega_i$, and $\partial\boldsymbol{q}_i/\partial\eta_i$ are calculated by differentiating Eq. (6). For example, if let J_{TCS} denotes the robotic Jacobian matrix at the center of $\{TCS\}$, then we have

$$\frac{\partial\boldsymbol{q}_i}{\partial\eta_i} = J_{TCS}^{-1}\begin{bmatrix}\boldsymbol{0}_{3\times1} \\ [0,\ 0,\ 1]^T\end{bmatrix} \tag{19}$$

For more details about the Jacobian and velocity kinematics of the robot, the reader can refer to Ref. [32].

Similarly, the first-order differential increment of ϕ_1^{ij} and ϕ_2^{ij} are

$$\Delta\phi_1^{ij} = \sum_{k=1}^{n}\frac{\partial\phi_1^{ij}}{\partial\theta_j^k}\left[\frac{\partial\theta_j^k}{\partial\lambda_k}, \frac{\partial\theta_j^k}{\partial\omega_k}, \frac{\partial\theta_j^k}{\partial\eta_k}\right][\Delta\lambda_k, \Delta\omega_k, \Delta\eta_k]^T \tag{20}$$

$$\Delta\phi_2^{ij} = \sum_{k=1}^{n}\frac{\partial\phi_2^{ij}}{\partial\theta_j^k}\left[\frac{\partial\theta_j^k}{\partial\lambda_k}, \frac{\partial\theta_j^k}{\partial\omega_k}, \frac{\partial\theta_j^k}{\partial\eta_k}\right][\Delta\lambda_k, \Delta\omega_k, \Delta\eta_k]^T \tag{21}$$

where $[\partial\theta_j^k/\partial\lambda_k,\ \partial\theta_j^k/\partial\omega_k,\ \partial\theta_j^k/\partial\eta_k]$ is the jth row of $[\partial\boldsymbol{q}_k/\partial\lambda_k,\ \partial\boldsymbol{q}_k/\partial\omega_k,\ \partial\boldsymbol{q}_k/\partial\eta_k]$. For brevity of the expressions, by denoting $\Delta\boldsymbol{\alpha} = [\Delta\lambda_1, \Delta\omega_1, \Delta\eta_1, \cdots, \Delta\lambda_n, \Delta\omega_n, \Delta\eta_n]$, Eqs. (20) and (21) are represented as

$$\Delta\phi_1^{ij} = [\partial\phi_1^{ij}/\partial\boldsymbol{\alpha}]^T\Delta\boldsymbol{\alpha} \tag{22}$$

$$\Delta\phi_2^{ij} = [\partial\phi_2^{ij}/\partial\boldsymbol{\alpha}]^T\Delta\boldsymbol{\alpha} \tag{23}$$

On this basis, near the nominal parameter $\boldsymbol{\alpha}_k = [\lambda_1^k, \omega_1^k, \eta_1^k, \cdots, \lambda_n^k, \omega_n^k, \eta_n^k]$, Φ_{smooth} can be approximated as a quadratic function

$$\begin{aligned}\Phi_{smooth} &\approx \sum_{i=1}^{n}\sum_{j=1}^{6} \left[k_1\left(\phi_1^{ij}(\boldsymbol{\alpha}_k) + \left[\partial\phi_1^{ij}/\partial\boldsymbol{\alpha}\right]^T\Delta\boldsymbol{\alpha}\right)^2 + k_2\left(\phi_2^{ij}(\boldsymbol{\alpha}_k) + \left[\partial\phi_2^{ij}/\partial\boldsymbol{\alpha}\right]^T\Delta\boldsymbol{\alpha}\right)^2\right]ds_i \\ &= \Delta\boldsymbol{\alpha}^T\boldsymbol{H}\Delta\boldsymbol{\alpha} + 2\boldsymbol{f}^T\Delta\boldsymbol{\alpha} + \sum_{i=1}^{n}\sum_{j=1}^{6}\left[k_1(\phi_1^{ij}(\boldsymbol{\alpha}_k))^2 + k_2(\phi_2^{ij}(\boldsymbol{\alpha}_k))^2\right]ds_i\end{aligned} \tag{24}$$

where

$$\boldsymbol{H} = \sum_{i=1}^{n}\left[\sum_{j=1}^{6}(k_1[\partial\phi_1^{ij}/\partial\boldsymbol{\alpha}][\partial\phi_1^{ij}/\partial\boldsymbol{\alpha}]^T + k_2[\partial\phi_2^{ij}/\partial\boldsymbol{\alpha}][\partial\phi_2^{ij}/\partial\boldsymbol{\alpha}]^T)\right]ds_i \tag{25}$$

$$\boldsymbol{f} = \sum_{i=1}^{n}\left[\sum_{j=1}^{6}(k_1\phi_1^{ij}(\boldsymbol{\alpha}_k)[\partial\phi_1^{ij}/\partial\boldsymbol{\alpha}] + k_2\phi_2^{ij}(\boldsymbol{\alpha}_k)[\partial\phi_2^{ij}/\partial\boldsymbol{\alpha}])\right]ds_i \tag{26}$$

Based on Eq. (24), problem **P2** near $\boldsymbol{\alpha}_k$ is approximated as a QP subproblem as follows, which can be solved with available solvers such as the function *quadprog* in MATLAB.

$$\textbf{P3}\quad \begin{array}{ll} \min\limits_{\Delta\boldsymbol{\alpha},\Delta\psi} & \Delta\boldsymbol{\alpha}^T\boldsymbol{H}\Delta\boldsymbol{\alpha} + 2\boldsymbol{f}^T\Delta\boldsymbol{\alpha} + k_3\Delta\psi \\ \text{s.t.} & \psi_k + \Delta\psi \ge -w_i(\boldsymbol{\alpha}_k) - [\partial w_i/\partial\lambda_i, \partial w_i/\partial\omega_i][\Delta\lambda_i, \Delta\omega_i]^T \\ & \boldsymbol{q}_{\min} \le \boldsymbol{q}_i(\boldsymbol{\alpha}_k) + \left[\frac{\partial\boldsymbol{q}_i}{\partial\lambda_i}, \frac{\partial\boldsymbol{q}_i}{\partial\omega_i}, \frac{\partial\boldsymbol{q}_i}{\partial\eta_i}\right][\Delta\lambda_i, \Delta\omega_i, \Delta\eta_i]^T \le \boldsymbol{q}_{\max} \\ & f_i(\omega_i^k) + f_i'(\omega_i^k)\Delta\omega_i \le \lambda_i^k + \Delta\lambda_i \le \pi/2 \\ & \omega_i^{low} \le \omega_i^k + \Delta\omega_i \le \omega_i^{high} \\ & i = 1, 2, \ldots, n \end{array}$$

where $\psi_k = -\min\limits_{i=1,\cdots,n} w_i(\boldsymbol{\alpha}_k)$. Since $[\partial w_i/\partial\lambda_i, \partial w_i/\partial\omega_i]$ can be obtained by the analytical approximation method proposed in Sec. 3.4,

numerical methods are only required for the calculation of machining width when constructing Problem **P3.** Considering that the approximation in **P3** only holds in the neighborhood of $\boldsymbol{\alpha}_k$, a parameter ζ is introduced to limit the range of the optimization variables. Then, problem **P3** can be reformulated into **P4**

$$
\mathbf{P4} \quad
\begin{aligned}
&\min_{\Delta\boldsymbol{\alpha},\Delta\psi} \quad \Delta\boldsymbol{\alpha}^T \boldsymbol{H}\Delta\boldsymbol{\alpha} + 2\boldsymbol{f}^T\Delta\boldsymbol{\alpha} + k_3\Delta\psi \\
&\text{s.t.} \quad \boldsymbol{A}_k \begin{bmatrix} \Delta\boldsymbol{\alpha} \\ \Delta\psi \end{bmatrix} \le \boldsymbol{b}_k \\
&\qquad \zeta\Delta\boldsymbol{\alpha}_{lower} \le \Delta\boldsymbol{\alpha} \le \zeta\Delta\boldsymbol{\alpha}_{upper}
\end{aligned}
$$

where $\boldsymbol{A}_k$ is a matrix, $\boldsymbol{b}_k$ is a vector, $\Delta\boldsymbol{\alpha}_{lower}$ and $\Delta\boldsymbol{\alpha}_{upper}$ are the lower and upper bounds of $\Delta\boldsymbol{\alpha}$, respectively. Since matrices $\boldsymbol{H}$ and $\boldsymbol{A}_k$ are both sparse, problem **P4** could be solved relatively quickly.

Based on the derived results earlier, an SQP-based algorithm is presented as follows to solve the posture optimization problem.

Algorithm (Sequential Quadratic Programming for Tool Posture Optimization)

Input: CC points $\{\boldsymbol{CC}_i | i = 1, 2, \cdots, n\}$; design surface; robot model; the radius of the cutter r; scallop height h; weighting parameters k_1, k_2 and k_3; initial solution $\boldsymbol{\alpha}_0$; threshold τ_1 and τ_2; lower bound function $\{f_i(\omega_i) | i = 1, 2, \cdots, n\}$; bounds $\Delta\boldsymbol{\alpha}_{lower}$ and $\Delta\boldsymbol{\alpha}_{upper}$.

Output: Optimized solution $\boldsymbol{\alpha}^*$.

Step 0:

(1) Set $k = 0$ and $\zeta = 1$;
(2) Compute $\boldsymbol{q}_i(\boldsymbol{\alpha}_0)$, $w_i(\boldsymbol{\alpha}_0)$ and $\Phi_{smooth}(\boldsymbol{\alpha}_0)$, $i = 1, \cdots, n$;
(3) Set $\psi_0 = -\min_{i=1,\cdots,n} w_i(\boldsymbol{\alpha}_0)$, $\Psi_0 = \Phi_{smooth}(\boldsymbol{\alpha}_0) + k_3\psi_0$.

Step 1:

(1) Compute $[\partial\boldsymbol{q}_i/\partial\lambda_i, \partial\boldsymbol{q}_i/\partial\omega_i, \partial\boldsymbol{q}_i/\partial\eta_i]$, $[\partial\phi_1^{ij}/\partial\boldsymbol{\alpha}]$, $[\partial\phi_2^{ij}/\partial\boldsymbol{\alpha}]$, and $[\partial w_i/\partial\lambda_i, \partial w_i/\partial\omega_i]$, for $i = 1, \cdots, n$, $j = 1. \cdots, 6$;
(2) Compute matrix $\boldsymbol{H}$ and vector $\boldsymbol{f}$ using Eqs. (25) and (26).

Step 2:

(1) Solve the QP problem **P4**;
(2) Update $\boldsymbol{\alpha}_{k+1} = \boldsymbol{\alpha}_k + \Delta\boldsymbol{\alpha}$;
(3) Compute $\boldsymbol{q}_i(\boldsymbol{\alpha}_{k+1})$, $w_i(\boldsymbol{\alpha}_{k+1})$ and $\Phi_{smooth}(\boldsymbol{\alpha}_{k+1})$, $i = 1, \cdots, n$;
(4) If constraints in Problem **P1** are not satisfied when $\boldsymbol{\alpha} = \boldsymbol{\alpha}_{k+1}$, then set $\zeta = 0.5\zeta$ and go to (1);
(5) Update $\psi_{k+1} = -\min_{i=1,\cdots,n} w_i(\boldsymbol{\alpha}_{k+1})$, $\Psi_{k+1} = \Phi_{smooth}(\boldsymbol{\alpha}_{k+1}) + k_3\psi_{k+1}$.

Step 3:

(1) If $\Psi_{k+1} < \Psi_k$, then set $k = k + 1$ and go to (2); Else go to (3);
(2) If $(\Psi_k - \Psi_{k+1})/|\Psi_k| > \tau_2$, then go to **Step 1**; Else exit and report $\boldsymbol{\alpha}^* = \boldsymbol{\alpha}_k$;
(3) If $\zeta > \tau_1$, then set $\zeta = 0.5\zeta$ and go to **Step 2**; Else exit and report $\boldsymbol{\alpha}^* = \boldsymbol{\alpha}_k$.

4 Calculations of the Machining Width and Its Derivative

The optimization process in Sec. 3.3 involves extensive calculations of machining width w_i and its derivative $[\partial w_i/\partial\lambda_i, \partial w_i/\partial\omega_i]$. Traditionally, w_i is obtained by numerical methods, and $[\partial w_i/\partial\lambda_i, \partial w_i/\partial\omega_i]$ can be calculated by numerical difference formula. To reduce the execution time of the algorithm, an efficient iterative method for calculating w_i, as well as an analytical approximation of $[\partial w_i/\partial\lambda_i, \partial w_i/\partial\omega_i]$, is proposed in this section based on CGA. It should be noted that the proposed methods could also be executed without CGA, i.e., by using the traditional vector algebra. However, the use of CGA allows for a more concise expression of formulas, better geometric intuition, and greater efficiency. The basic elements, operations, and other foundations of CGA used in this section can be found in Appendix.

4.1 Calculation of the Machining Width. According to Eq. (7), the key to calculating w_i is to find two points $\boldsymbol{p}_a$ and $\boldsymbol{p}_b$ on BTC with a distance h from the design surface. Given a point $\boldsymbol{q}$ on a surface, the squared distance from a point $\boldsymbol{p}$ near $\boldsymbol{q}$ to the surface can be quadratically approximated by the squared distance from $\boldsymbol{p}$ to the tangent plane of the surface at $\boldsymbol{q}$ [33]. On this basis, an iterative method to find the point on BTC with a distance h from the design surface is proposed as follows.

As shown in Fig. 5, suppose that the current iteration point is $\boldsymbol{p}^k$ and the nearest point to it on the design surface is $\boldsymbol{q}^k$. Denote the offset plane of the tangent plane at $\boldsymbol{q}^k$ with an offset distance h as T_h^k in CGA, which can be evaluated by

$$T_h^k = P(\boldsymbol{q}^k + h\boldsymbol{n}^k) \wedge \boldsymbol{t}_1^k \wedge \boldsymbol{t}_2^k \wedge e_\infty \tag{27}$$

where $P(\cdot)$ is the representation of any point in CGA, defined in Appendix, $\boldsymbol{n}^k$ is the normal of the design surface at $\boldsymbol{q}^k$, and $\boldsymbol{t}_1^k$ $\boldsymbol{t}_2^k$ are two independent tangent vectors of the design surface at $\boldsymbol{q}^k$. BTC can be represented in CGA as

$$C_{BTC} = P(\boldsymbol{p}_1) \wedge P_2(\boldsymbol{p}_2) \wedge P_3(\boldsymbol{p}_3) \tag{28}$$

where $\{\boldsymbol{p}_i | i = 1, 2, 3\}$ are three points on BTC. The intersection of BTC and T_h^k is calculated by

$$P(\boldsymbol{p}_1^k) \wedge P(\boldsymbol{p}_2^k) = [C_{BTC}^* \wedge (T_h^k)^*]^* \tag{29}$$

where $\boldsymbol{p}_1^k$ and $\boldsymbol{p}_2^k$ are the intersection points of BTC and plane T_h^k. The point that is closer to $\boldsymbol{p}^k$ among the two points $\boldsymbol{p}_1^k$ and $\boldsymbol{p}_2^k$ is taken as the next iteration point $\boldsymbol{p}^{k+1}$.

To obtain points $\boldsymbol{p}_a$ and $\boldsymbol{p}_b$, two initial iteration points $\boldsymbol{p}_a^0$ and $\boldsymbol{p}_b^0$ need to be fed into the above iterative process. Denote the offset plane of the tangent plane at $\boldsymbol{CC}_i$ with an offset distance h as $T_h^{CC_i}$, then a reasonable choice for the initial iteration points is the intersection of BTC and $T_h^{CC_i}$, as shown in Fig. 6. The calculation process is similar to Eq. (29)

In most cases, the proposed iterative approach can converge to the desired result at a much faster rate than the general numerical algorithms for solving equations. However, there is also a small possibility that the method fails to converge. Thus, when the case

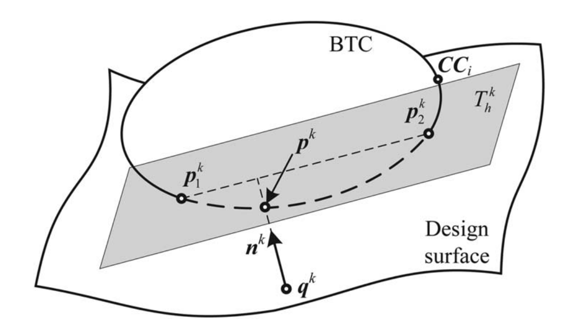


Fig. 5 Computation of p_a and p_b

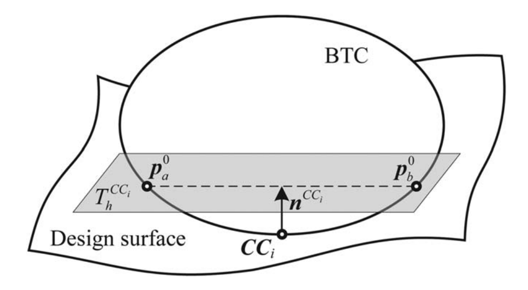


Fig. 6 Computation of the initial iteration points. n^{CC_i} is the normal vector of the design surface at CC_i.

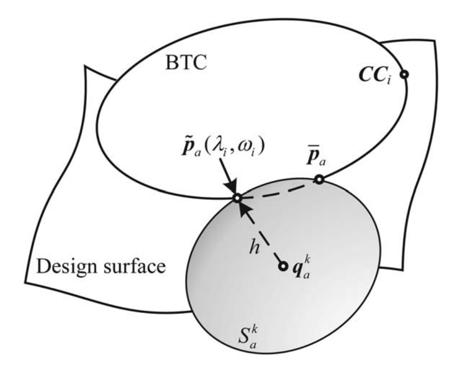


Fig. 7 Approximation of p_a

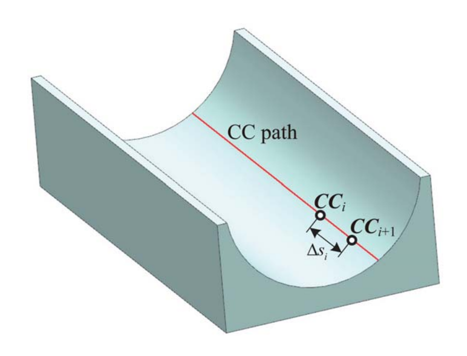


Fig. 8 CC Path of Example 1

$||\boldsymbol{p}^k - \boldsymbol{q}^k|| - h| < ||\boldsymbol{p}^{k+1} - \boldsymbol{q}^{k+1}|| - h|$ is encountered, other numerical methods such as the bisection method need to be adopted.

4.2 Analytical Approximation of the Derivatives of the Machining Width. According to Eq. (7), the key to obtaining $[\partial w_i/\partial\lambda_i, \partial w_i/\partial\omega_i]$ is to calculate the derivatives of $\boldsymbol{p}_a$ and $\boldsymbol{p}_b$, i.e., $[\partial\boldsymbol{p}_a/\partial\lambda_i, \partial\boldsymbol{p}_a/\partial\omega_i]$ and $[\partial\boldsymbol{p}_b/\partial\lambda_i, \partial\boldsymbol{p}_b/\partial\omega_i]$, where $\boldsymbol{p}_a$ and $\boldsymbol{p}_b$ are the two points on BTC at a distance h from the design surface. According to Sec. 4.1, these two points cannot be expressed in an analytical form; thus, the derivatives of them cannot be calculated analytically. To improve the efficiency, an analytical method to approximate the derivatives is proposed in this section. The approximation of $[\partial\boldsymbol{p}_a/\partial\lambda_i, \partial\boldsymbol{p}_a/\partial\omega_i]$ is presented as follows, while the derivatives of $\boldsymbol{p}_b$ can be obtained in the same way.

Suppose the current iteration solution of the SQP Algorithm in Sec. 3.3 is $\boldsymbol{\alpha}_k = [\lambda_1^k, \omega_1^k, \eta_1^k, \cdots, \lambda_n^k, \omega_n^k, \eta_n^k]$. At $\boldsymbol{CC}_i$, denote the point on the design surface that is nearest to $\boldsymbol{p}_a(\lambda_i^k, \omega_i^k)$ as $\boldsymbol{q}_a^k$. While $\boldsymbol{p}_a(\lambda_i, \omega_i)$ is the point at a distance h from the design surface, the point $\tilde{\boldsymbol{p}}_a(\lambda_i, \omega_i)$ on BTC at a distance h from $\boldsymbol{q}_a^k$ is used as an approximation of $\boldsymbol{p}_a$, as shown in Fig. 7. Note that $\tilde{\boldsymbol{p}}_a(\lambda_i^k, \omega_i^k) = \boldsymbol{p}_a(\lambda_i^k, \omega_i^k)$.

In CGA, the sphere with center $\boldsymbol{q}_a^k$ and radius h can be represented as

$$S_a^k = (P(\boldsymbol{q}_a^k) - h^2 e_\infty/2)^* \tag{30}$$

As shown in Fig. 7, $\tilde{\boldsymbol{p}}_a$ can be regarded as the intersection of C_{BTC} and S_a^k, and it is obtained by

$$P[\tilde{\boldsymbol{p}}_a(\lambda_i, \omega_i)] \wedge P[\bar{\boldsymbol{p}}_a(\lambda_i, \omega_i)] = [C_{BTC}^*(\lambda_i, \omega_i) \wedge S_a^*]^* \tag{31}$$

where $\tilde{\boldsymbol{p}}_a$ satisfies $\tilde{\boldsymbol{p}}_a(\lambda_i^k, \omega_i^k) = \boldsymbol{p}_a(\lambda_i^k, \omega_i^k)$, and $\bar{\boldsymbol{p}}_a$ is another intersection point.

Since $\tilde{\boldsymbol{p}}_a(\lambda_i, \omega_i)$ is an analytical function, the derivative $[\partial\tilde{\boldsymbol{p}}_a/\partial\lambda_i, \partial\tilde{\boldsymbol{p}}_a/\partial\omega_i]$ can be calculated analytically, which is used as the approximation of $[\partial\boldsymbol{p}_a/\partial\lambda_i, \partial\boldsymbol{p}_a/\partial\omega_i]$. The simulations and experiments in Sec. 5 show that the proposed approximation method has a relatively small error, while the computational efficiency is much higher than that of the numerical difference method.

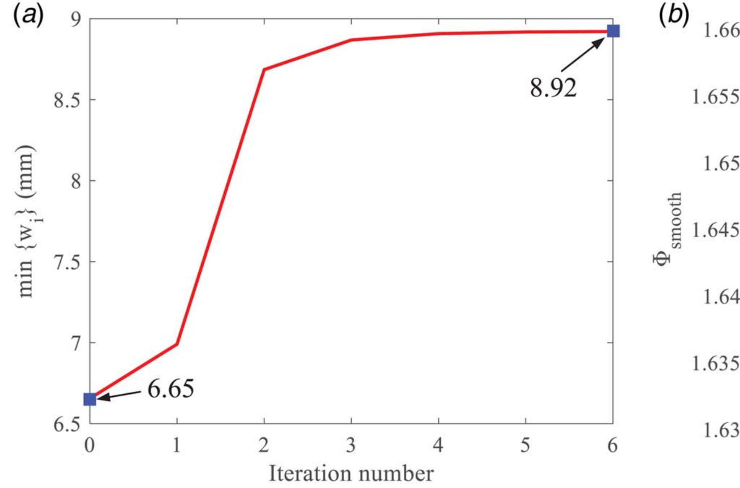
Table 1 DH Parameters of UR 10

Joint	θ	a (m)	d (m)	α (rad)
Joint 1	θ_1	0	0.1273	$\pi/2$
Joint 2	θ_2	−0.612	0	0
Joint 3	θ_3	−0.5723	0	0
Joint 4	θ_4	0	0.163941	$\pi/2$
Joint 5	θ_5	0	0.1157	$-\pi/2$
Joint 6	θ_6	0	0.0922	0

5 Simulations and Experiments

To verify the validity of the presented approach, it is performed in the milling of three different surfaces.

5.1 Initial Solution Generation. The initial solution for SQP is obtained by the following traditional two-step method. Firstly, the tool axis direction, which is represented by the inclination angle λ_i and tilt angle ω_i, is calculated sequentially using the



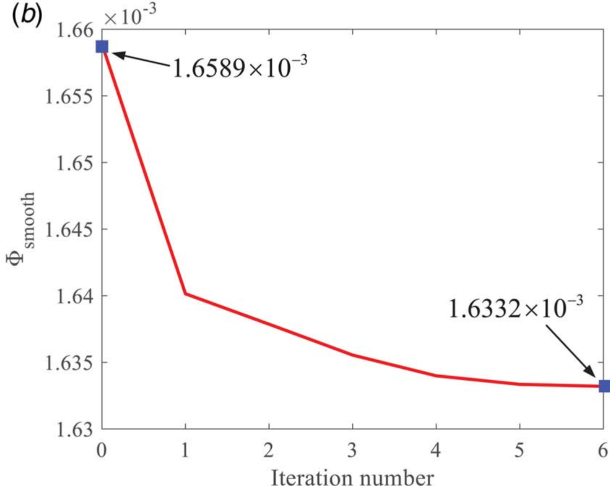


Fig. 9 Toolpath optimization process in Example 1: (a) the variation of min{w_i} and (b) the variation of Φ_{smooth}

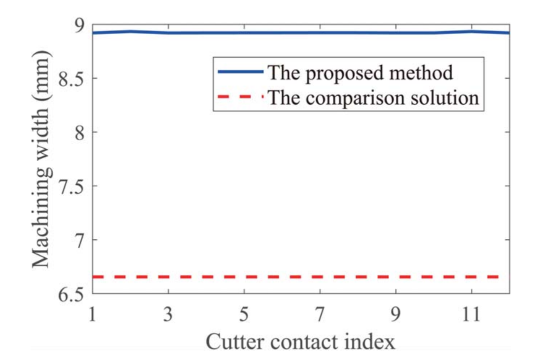


Fig. 10 The machining width in Example 1

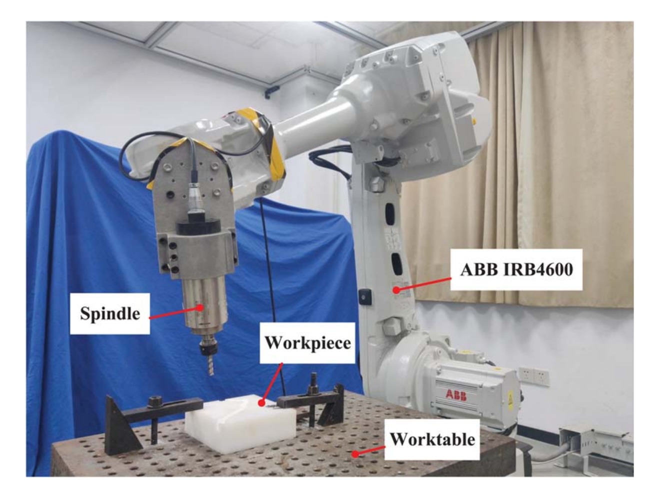


Fig. 11 The experimental platform of robot milling

rolling ball method [25] at each CC point on the CC path. This method can ensure that no local or rear gouging occurs and that the machining width is relatively large in most cases. Secondly, based on the tool axis directions, the graph-based method proposed in Ref. [6] is used to optimize the robot redundancy $\{\lambda_i\}$ at all CC points simultaneously. The optimization objective used in this process is set as $\tilde{\Phi} = \sum_{i=2}^{n}\sum_{j=1}^{6}\left[\left(\theta_j^i - \theta_j^{i-1}\right)/\Delta s_{i-1}\right]^2 \Delta s_{i-1}$, which is an approximation of the term $\int_0^l \sum_{j=1}^{6} (d\theta_j/ds)^2 ds$ in the smoothness index shown in Eq. (9). In the following sections, the initial solution is also used as the comparison solution.

Table 2 DH Parameters of ABB IRB 4600-60

Joint	θ	a (m)	d (m)	α (rad)
Joint 1	θ_1	0.175	0.495	$-\pi/2$
Joint 2	θ_2	0.9	0	0
Joint 3	θ_3	0.175	0	$-\pi/2$
Joint 4	θ_4	0	0.96	$\pi/2$
Joint 5	θ_5	0	0	$\pi/2$
Joint 6	θ_6	0	0.135	0

5.2 Example 1. In this section, the validity of the proposed approach is demonstrated by a simulation of toolpath optimization during the milling of a simple cylindrical surface with a radius of 50 mm, which appears in Ref. [15]. The CC path consists of 12 CC points, and it is shown in Fig. 8. The feed direction is always the same as the cylinder axis direction. For $i = 1, \cdots, 11$, $\Delta s_i =$ 20mm. The radius of BTC is 6 mm. The industrial robot (IR) used is the Universal Robot UR10, and its Denavit–Hartenberg (DH) convention parameters are shown in Table 1. The scallop height is 0.01 mm. The three weighting parameters are $k_1 = 0.5$, $k_2 = 10^4$, and $k_3 = 1$. The thresholds are set as $\tau_1 = 10^{-6}$ and $\tau_2 = 10^{-4}$.

The toolpath optimization method converges quickly within six iterations, and the iteration process is presented in Fig. 9. It depicts that the minimum machining width among all the CC points increases from 6.65 mm to 8.92 mm, while the performance index of the smoothness decreases from 1.6589×10^{-3} to 1.6332×10^{-3}. The effectiveness of the proposed method is validated. As illustrated in Fig. 10, compared with the comparison solution, the machining width at every CC point obtained by the proposed method is improved in this example. In fact, the obtained machining width is rather close to the optimal machining width 9.07 mm [19].

5.3 Example 2. To verify the validity of the proposed method, physical robot milling experiments are given in this section. The experimental platform is illustrated in Fig. 11. The IR used is ABB IRB 4600-60, and Table 2 gives its DH parameters. The radius of BTC is 5 mm. The position and orientation of $\{TCS\}$ w.r.t. $\{FCS\}$, and $\{WCS\}$ w.r.t. $\{BCS\}$ are measured and shown in Table 3.

As Fig. 12 shows, the design surface considered in the experiments is a bicubic Bezier surface which appears in Ref. [19]. Its control points represented in $\{WCS\}$ are

$$CP = \begin{bmatrix} (0,0,15) & (0,50,65) & (0,102,2.3) & (0,150,15) \\ (76,0,65) & (50,76,-10) & (50,102,65) & (89,150,53) \\ (102,0,-5.5) & (102,102,40) & (102,76,-10) & (127,150,15) \\ (150,0,15) & (150,12.7,-10) & (150,76,48) & (150,1150,15) \end{bmatrix} \tag{32}$$

The CC path consists of 104 CC points. The feed direction at every CC point is chosen as the minimum curvature direction of the surface since the machining width along this direction is always relatively large [14]. For $i = 1, \cdots, 103$, Δs_i is around

Table 3 Parameters of frames {*TCS*} and {*WCS*} in Example 2

Frame	Position [x, y, z] (mm)	Quaternion [$q1$, $q2$, $q3$, $q4$]
TCS	[322.198, 191.753, 69.1797]	[0.68301, 0.18301, −0.68301, 0.18301]
WCS	[1056.25, 253.88, 598.46]	[1, 0, 0, 0]

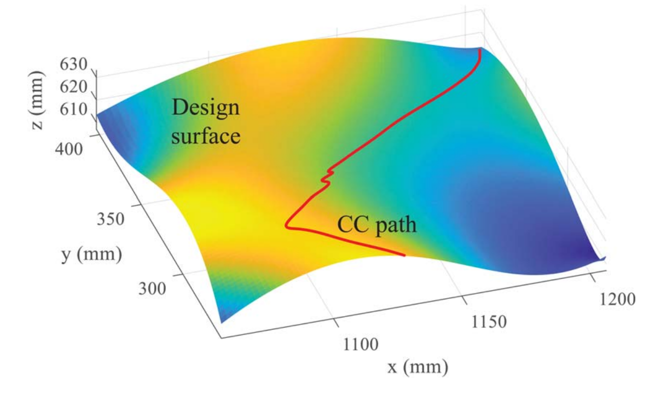


Fig. 12 Design surface and CC path represented in {*BCS*} of Example 2

2mm. The scallop height is 0.01mm. The three weighting parameters are $k_1 = k_2 = 10^{-4}$, and $k_3 = 1$, and the thresholds are set as $\tau_1 = \tau_2 = 10^{-5}$.

The toolpath optimization method converges within 24 iterations, and the iteration process is presented in Fig. 13. It illustrates that the minimum machining width among all the CC points increases by 108.45% from 1.42mm to 2.96mm, while the performance index of the smoothness decreases by 89.93% from 1.43×10^{-5} to 1.44×10^{-6}. The good performance and fast convergence speed of the optimization algorithm are thus proven. In practice, by appropriately increasing the termination threshold τ_1 and τ_2, the number of iterations can be further reduced at the cost of a slight loss of optimization objective.

The comparison of the machining width at each CC point is shown in Fig. 14. The minimum machining width with the comparison solution occurs at the 101st CC point with a size of 1.42 mm, while that with the proposed method occurs at the 1st CC point with a size of 2.96 mm. It is obvious that the minimum machining width is improved.

The six joint paths of the robot are illustrated in Fig. 15, and it shows that the proposed approach facilitates smoother joint paths. For Joint 2, Joint 4, and Joint 6, the smoothness index ϕ_1^{ij} and ϕ_2^{ij} are presented in Fig. 16. It shows that the two types of smoothness index of the comparison solution have relatively much larger values in many positions, while those of the proposed method keep a small value.

Note that the CC points at which the robot joint paths obtained by the comparison solution fluctuate drastically correspond to the sharp corners on the CC path shown in Fig. 12. Since the tool axis direction is generated without considering the path smoothness during the generation of the comparison solution, the fluctuations at these CC points cannot be eliminated even if the smoothness is considered during the redundancy optimization. The results

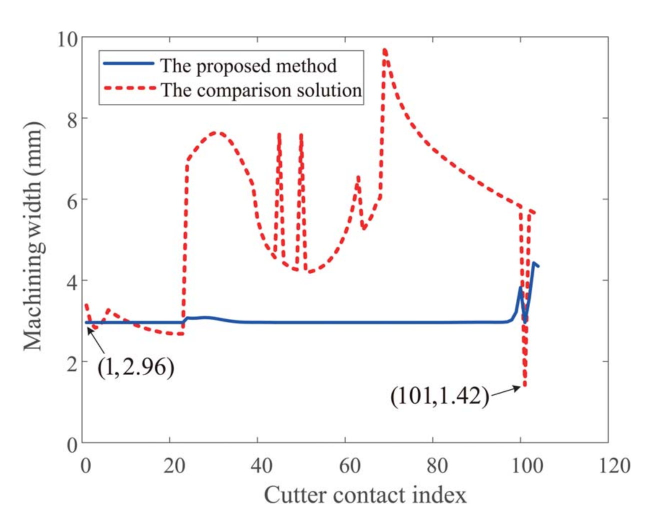


Fig. 14 Machining width in Example 2

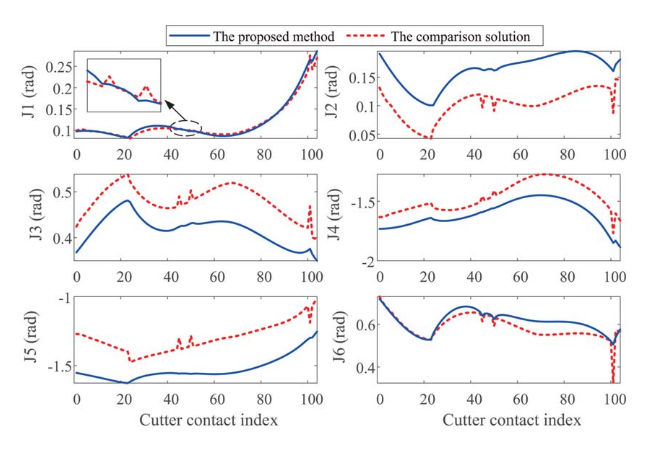


Fig. 15 Joint position of the robot in Example 2. "*Ji*" is short for "Joint *i*".

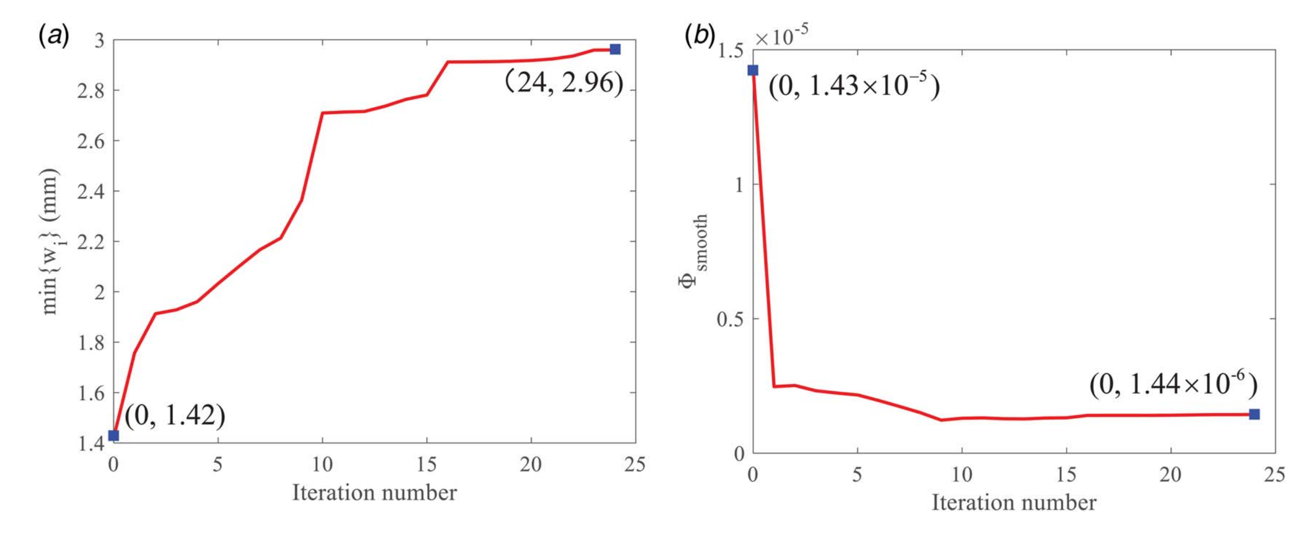


Fig. 13 Toolpath optimization process in Example 2: (*a*) the variation of min{w_i} and (*b*) the variation of Φ_{smooth}

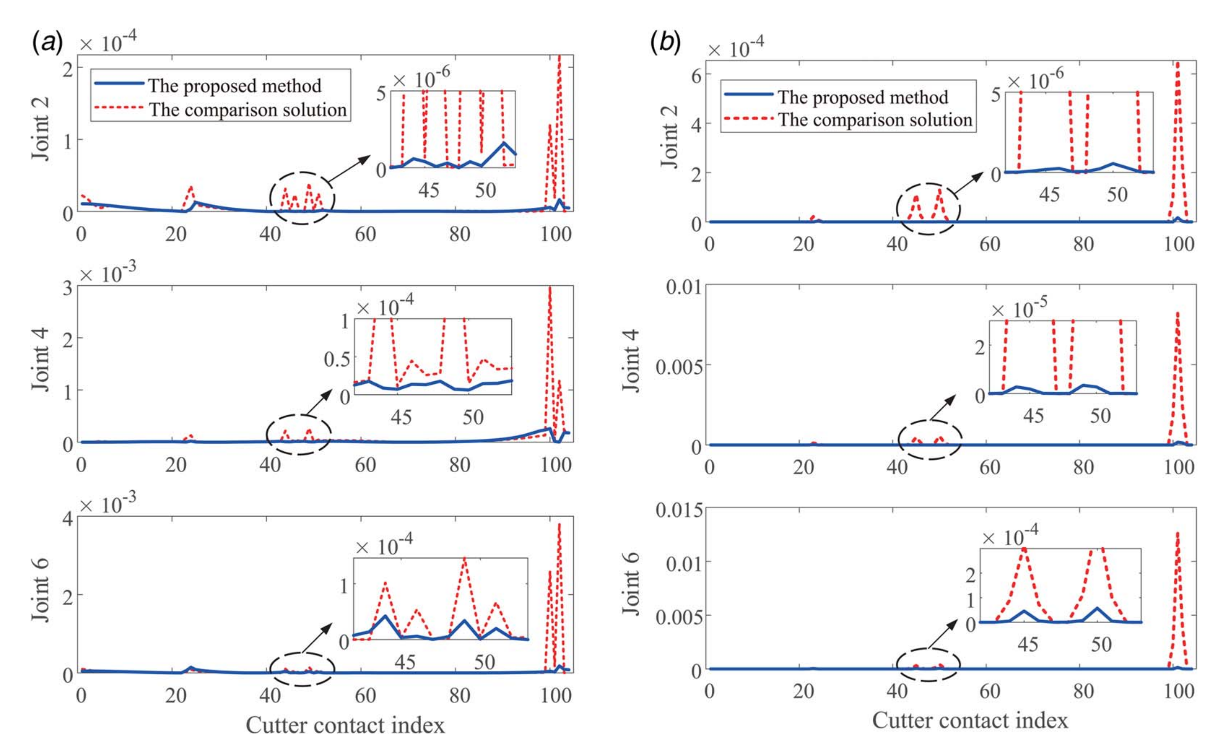


Fig. 16 Smoothing index of Joint 2, Joint 4, and Joint 6 in Example 2: (a) the smoothing index $(\phi_1^{ij})^2$ and (b) the smoothing index $(\phi_2^{ij})^2$

demonstrate better robustness of the proposed method compared to the conventional two-step method of sequential planning of tool axis direction and robot redundancy.

To test the validity of the analytical approximation presented in Sec. 4.2, the derivatives of the machining width are calculated using the approximation method and the numerical difference method, respectively, in the optimization process. On a 2.90 GHz Intel Core i7-10700 CPU, both methods are executed with MATLAB R2022b. During the optimization, a total of 14,829 calculations of the derivatives of the machining width w.r.t. the two parameters are performed, and a comparison of the calculation time of the two methods is shown in Table 4. It is clear that the approximation method takes less than one-tenth of the computation time of the numerical method, which makes a difference in the efficiency of the SQP algorithm. The error of the approximate solutions relative to the numerical solutions is also calculated. The results show that 94.25% of the approximate solutions have an error of less than 1%, which indicates that the proposed approximation is reasonable.

To verify the reliability of the optimization results, milling simulations are performed in VERICUT. The distribution of machining errors is obtained using the automatic comparison module in the software, and the results are illustrated in Fig. 17. According to the reports given by VERICUT, there is no overcutting greater than 1×10^{-5}mm in the workpiece after machining with both toolpaths, which verifies that the proposed algorithm can avoid local and rear gouges. Besides, the machining width distributions in Fig. 17 are generally consistent with the calculated results in Fig. 14.

RobotStudio, an offline programming software, is used to conduct the simulation of the milling process. The feedrate of the tooltip is set to 10 mm/s, and the actual feedrate recorded is illustrated in Fig. 18. It shows that the fluctuations in the feedrate curve of the proposed method are relatively small, which is mainly due to the smoothness of the joint paths. The time for machining with the proposed approach is 21.77 s, while that of the comparison method is 24.31 s. It means that the milling efficiency of the CC path is improved by 10.46%. Considering that the 108.45% improvement of the minimum machining width leads to much fewer CC paths required for milling the entire design surface, the cycle time for milling the entire design surface is expected to be reduced by more than 10.46%.

The material for milling is Delrin. Before the experiments, rough machining of the cubic workpiece blanks was carried out. The cutting depth is set to 1 mm, and the spindle speed is 12,000 r/min. The machined workpieces are presented in Fig. 19. The workpiece surface obtained using the comparison method is relatively rougher, while the surface quality of the workpiece obtained with the proposed optimization method is higher. The portable surface roughness tester is used to measure the surface roughness locally. It shows that for Area a in Fig. 19, the arithmetic mean roughness (Ra) with the proposed method is 0.325μm, while that of the surface machined with the comparison solution is 3.805μm. It shows that the smoothness of the machined surface is greatly improved.

Table 4 The calculation time of the derivatives of the machining width

Method	Total time (s)	Average time (s)
The approximation method	32.77	0.0022
The numerical difference method	369.06	0.0249

5.4 Example 3. To further verify the effect of the path smoothness on improving the machined surface quality, S-shaped curve milling experiments are conducted on a flat surface in this section. The CC path is shown in Fig. 20, which consists of 300 CC points. The robot and the tool configuration are the same as in Example 2.

The scallop height is 0.1mm. The three weighting parameters are $k_1 = k_2 = 100$, and $k_3 = 1$, and the thresholds are set to $\tau_1 = \tau_2 = 0.1$. The joint paths of the robot before and after optimization are shown in Fig. 21, and the path smoothness index $\sum_{j=1}^{6} [k_1(\phi_1^{ij})^2 + k_2(\phi_2^{ij})^2]$ is presented in Fig.22. It is obvious that the path with the proposed approach is smoother than that with the comparison solution at almost every CC point.

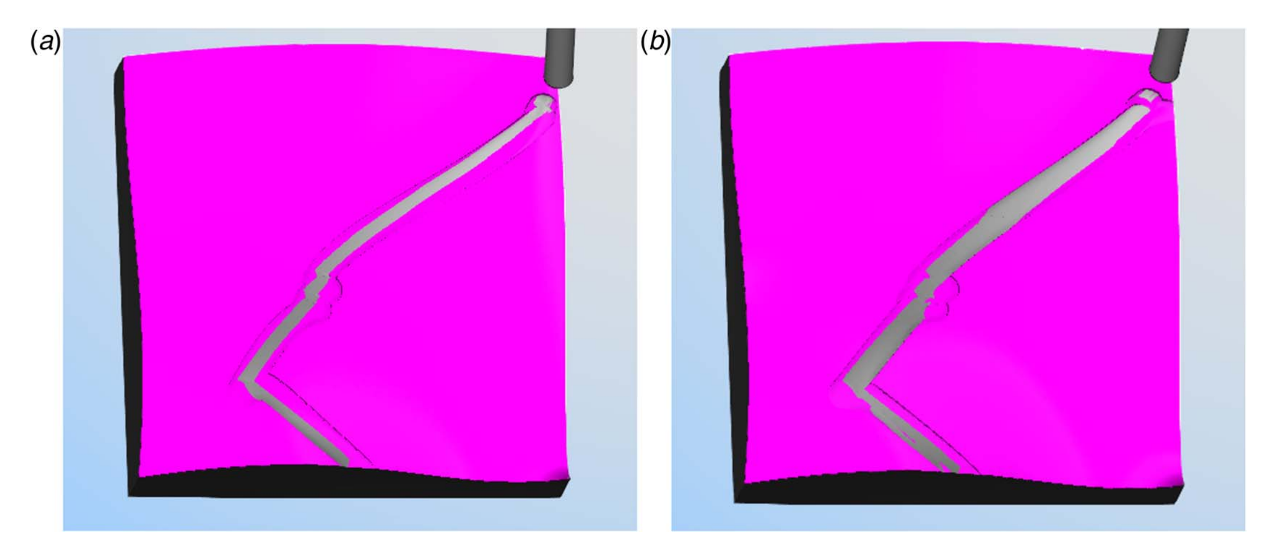


Fig. 17 Milling simulation results in VERICUT. Workpiece surfaces with an overcut of less than 1×10^{-5}mm and an undercut of less than 0.01 mm are shown in grey, and surfaces with an undercut of greater than 0.01 mm are shown in magenta: (*a*) the proposed method and (*b*) the comparison solution.

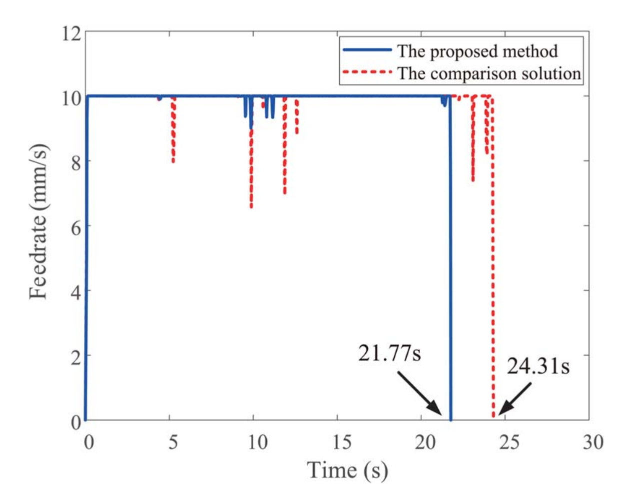


Fig. 18 Actual feedrate curves of the tooltip in the RobotStudio virtual simulation

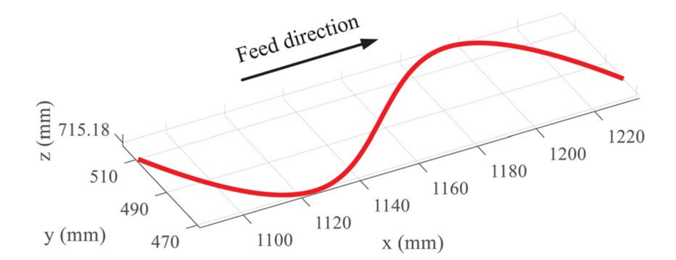
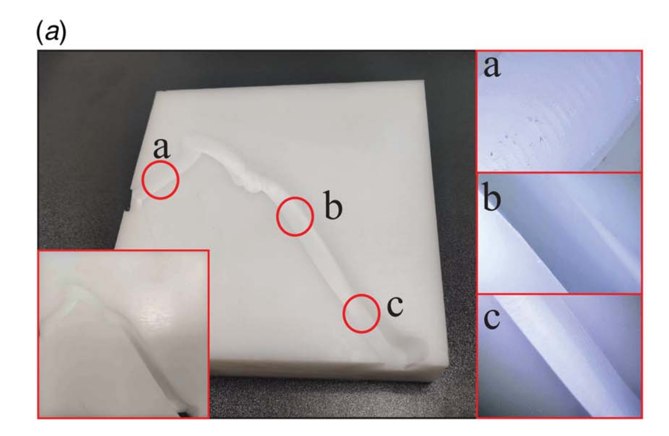


Fig. 20 CC Path in Example 3

The workpiece material used is wood. The cutting depth is set to 1 mm, and the spindle speed is 12,000 r/min. Figure 23 shows that the workpiece surface with the optimized toolpath is smoother at almost every position on the CC path, while that obtained using the comparison toolpath has more obvious tool marks. Considering the extremely small differences in joint angles of the two results near the CC points marked by the vertical dashed lines in Fig. 21, the dynamical performance (e.g., stiffness) of the robot induced by its configuration at these positions can be considered almost the same. Thus, it can be concluded that the surface quality improvement of the workpiece near these points is mainly determined by the increase in robot path smoothness. The results of the experiment validate that the proposed optimization method ameliorates the machining quality of robotic milling.



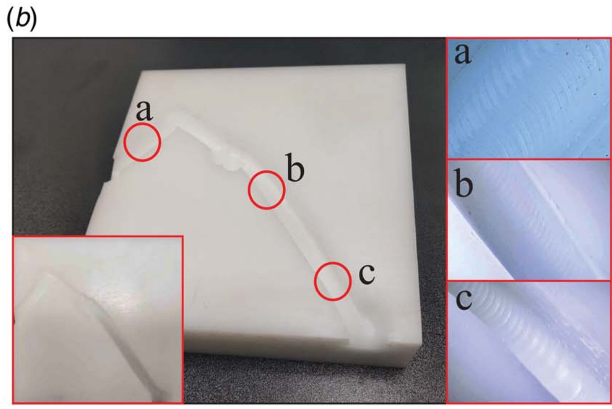


Fig. 19 The machined workpieces in Example 2: (*a*) the proposed method and (*b*) the comparison solution

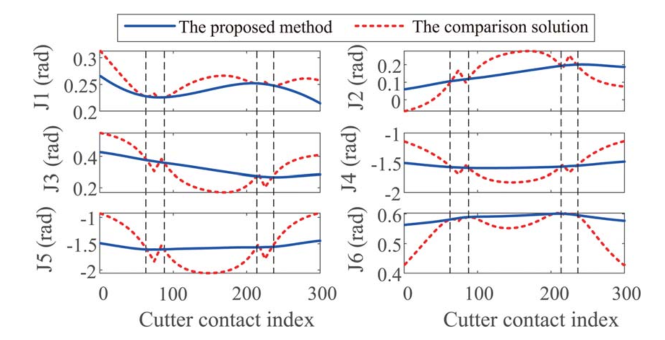


Fig. 21 Joint paths in Example 3

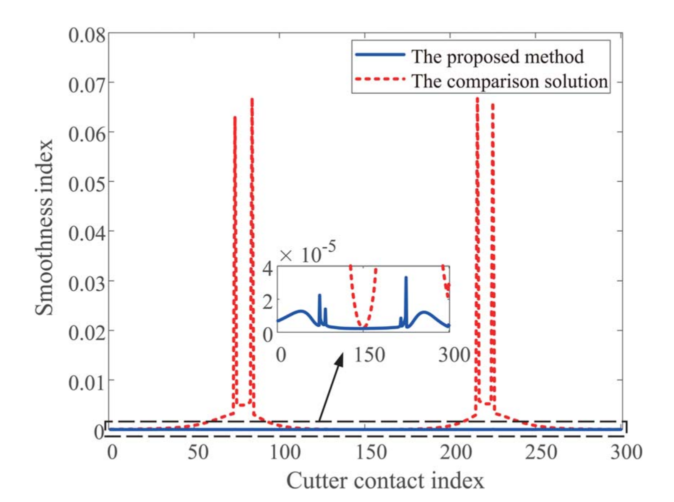


Fig. 22 Path smoothness index in Example 3

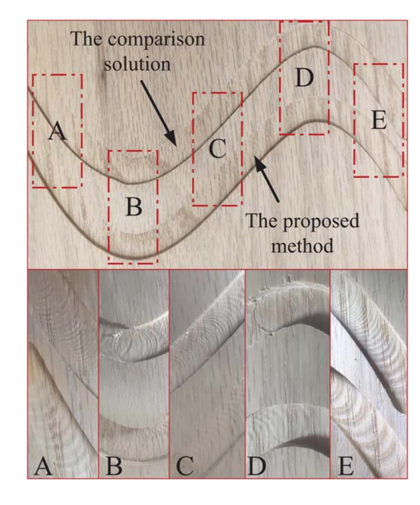


Fig. 23 Machined workpiece in Example 3

6 Conclusion

This paper proposes an effective posture optimization method for robotic flat-end milling. The inclination, tilt angles, and robot redundancy at every CC point on a path are optimized simultaneously to generate a robot path satisfying the joint constraints and gouging-free constraints, while improving the path smoothness and machining width. An efficient method based on SQP is proposed to solve this nonconvex problem. The machining width is calculated by a CGA-based iteration method, and its derivatives are approximated by an analytical method. The developed approach is tested with three surface examples, validating the effectiveness and efficiency of the method. Compared with the traditional method, the proposed approach could generate a much smoother toolpath and significantly increase the machining width, resulting in improved surface conditions and processing productivity.

The proposed framework can also take other criteria such as the robot stiffness as constraints or objective functions. In future work, the generation of the feed direction and the adjacent tool paths for freeform surface machining is expected to be included in the proposed framework.

Acknowledgment

This work was partially supported by the National Natural Science Foundation of China (Grant Nos. 51935010 and 51822506).

Conflict of Interest

There are no conflicts of interest.

Data Availability Statement

The data sets generated and supporting the findings of this article are obtainable from the corresponding author upon reasonable request.

Appendix: Foundations of Conformal Geometric Algebra

CGA is a widely used version of geometric algebra [34,35], which allows uniform algebraic representation and operations for classical geometry. It is distinguished by the richness of its basic geometric elements and by the coordinate independence and elegance of the operations. For more details of CGA, the readers are referred to Ref. [36].

Assuming that the Euclidean space is spanned by vectors $\{e_1, e_2, e_3\}$, the basic vectors of its corresponding conformal space are $\{e_0, e_1, e_2, e_3, e_\infty\}$, where e_0 and e_∞ represent the origin and infinity of three-dimensional space, respectively. The representation of any point $\boldsymbol{p} = [p_x, p_y, p_z]^T$ in CGA has the following form:

$$P(\boldsymbol{p}) = e_0 + p_x e_1 + p_y e_2 + p_z e_3 + \|\boldsymbol{p}\|^2 e_\infty/2 \tag{A1}$$

One of the most important products in geometric algebra is the geometric product. For two vectors $a = a_0e_0 + a_1e_1 + a_2e_2 + a_3e_3 + a_\infty e_\infty$ and $b = b_0e_0 + b_1e_1 + b_2e_2 + b_3e_3 + b_\infty e_\infty$, the geometric product of them is defined as Eq. (A2)

$$ab = a \wedge b + a \cdot b \tag{A2}$$

where the two terms $a \wedge b$ and $a \cdot b$ denote the outer product and inner product, respectively. The outer product is also called the wedge product, and the outer product of two vectors is a bivector. The inner product in CGA is defined as identical to the dot product in classical vector algebra, but special attention is needed when vector e_0 or e_∞ is involved. Specifically, the basic rules of the inner product are

$$\begin{aligned} &e_0 \cdot e_0 = 0,\ e_0 \cdot e_\infty = -1,\ e_0 \cdot e_i = 0 \\ &e_i \cdot e_j = \delta_{i,j},\ e_i \cdot e_\infty = 0,\ e_\infty \cdot e_\infty = 0 \end{aligned} \tag{A3}$$

where $i, j \in \{1, 2, 3\}$, and

$$\delta_{i,j} = \begin{cases} 1,\ i = j \\ 0,\ i \neq j \end{cases} \tag{A4}$$

The dual of any element in the conformal space is defined by

$$(\cdots)^{*} = -(\cdots)\cdot(e_0 \wedge e_1 \wedge e_2 \wedge e_3 \wedge e_\infty) \tag{A5}$$

References

[1] Xiong, G., Ding, Y., and Zhu, L., 2019, "Stiffness-Based Pose Optimization of an Industrial Robot for Five-Axis Milling," Rob. Comput. Integr. Manuf., **55**, pp. 19–28.

[2] Liao, Z., Wang, Q. H., Xie, H., Li, J. R., Zhou, X., and Hua, P., 2022, "Optimization of Robot Posture and Workpiece Setup in Robotic Milling With Stiffness Threshold," IEEE/ASME Trans. Mechatron., **27**(1), pp. 582–593.

[3] Chen, Q., Zhang, C., Hu, T., Zhou, Y., Ni, H., and Xue, X., 2022, "Posture Optimization in Robotic Machining Based on Comprehensive Deformation Index Considering Spindle Weight and Cutting Force," Rob. Comput. Integr. Manuf., **74**, p. 102290.

[4] Léger, J., and Angeles, J., 2016, "Off-Line Programming of Six-Axis Robots for Optimum Five-Dimensional Tasks," Mech. Mach. Theory, **100**, pp. 155–169.

[5] Lin, J., Ye, C., Yang, J., Zhao, H., Ding, H., and Luo, M., 2022, "Contour Error-Based Optimization of the End-Effector Pose of a 6 Degree-of-Freedom Serial Robot in Milling Operation," Rob. Comput. Integr. Manuf., **73**, p. 102257.

[6] Peng, J. F., Ding, Y., Zhang, G., and Ding, H., 2020, "Smoothness-Oriented Path Optimization for Robotic Milling Processes," Sci. China Technol. Sci., **63**(9), pp. 1751–1763.

[7] Lu, Y. A., Tang, K., and Wang, C. Y., 2021, "Collision-Free and Smooth Joint Motion Planning for Six-Axis Industrial Robots by Redundancy Optimization," Rob. Comput. Integr. Manuf., **68**, p. 102091.

[8] Lu, L., Zhang, J., Tian, X., Han, J., and Wang, H., 2021, "Tool Path Optimization for Robotic Surface Machining by Using Sampling-Based Motion Planning Algorithms," ASME J. Manuf. Sci. Eng., **143**(1), p. 011002.

[9] Lu, L., Han, J., Dong, F., Ding, Z., Fan, C., Chen, S., Liu, H., and Wang, H., 2022, "Joint-Smooth Toolpath Planning by Optimized Differential Vector for Robot Surface Machining Considering the Tool Orientation Constraints," IEEE/ASME Trans. Mechatron., **27**(4), pp. 2301–2311.

[10] Liao, Z. Y., Li, J. R., Xie, H. L., Wang, Q. H., and Zhou, X. F., 2020, "Region-Based Toolpath Generation for Robotic Milling of Freeform Surfaces With Stiffness Optimization," Rob. Comput. Integr. Manuf., **64**, p. 101953.

[11] Li, Z., Peng, F., Yan, R., Tang, X., Xin, S., and Wu, J., 2022, "A Virtual Repulsive Potential Field Algorithm of Posture Trajectory Planning for Precision Improvement in Robotic Multi-Axis Milling," Rob. Comput. Integr. Manuf., **74**, p. 102288.

[12] Vickers, G. W., and Quai, K. W., 1989, "Ball-Mills Versus End-Mills for Curved Surface Machining," ASME J. Manuf. Sci. Eng., **111**(1), pp. 22–26.

[13] Fard, B., and Feng, M. J., and Y, H., 2009, "Effect of Tool Tilt Angle on Machining Strip Width in Five-Axis Flat-End Milling of Free-Form Surfaces," Int. J. Adv. Manuf. Technol., **44**(3–4), pp. 211–222.

[14] Chiou, J. C. J., and Lee, Y. S., 2005, "Optimal Tool Orientation for Five-Axis Tool-end Machining by Swept Envelope Approach," ASME J. Manuf. Sci. Eng., **127**(4), pp. 810–818.

[15] Fard, B., and Feng, M. J., and Y, H., 2010, "Effective Determination of Feed Direction and Tool Orientation in Five-Axis Flat-end Milling," ASME J. Manuf. Sci. Eng., **132**(6), p. 061011.

[16] Gong, H., Cao, L. X., and Liu, J., 2008, "Second Order Approximation of Tool Envelope Surface for 5-Axis Machining With Single Point Contact," Comput.-Aided Des., **40**(5), pp. 604–615.

[17] Gong, H., Fang, F. Z., Hu, X. T., Cao, L. X., and Liu, J., 2010, "Optimization of Tool Positions Locally Based on the BCELTP for 5-Axis Machining of Free-Form Surfaces," Comput.-Aided Des., **42**(6), pp. 558–570.

[18] Fan, J., and Ball, A., 2014, "Flat-end Cutter Orientation on a Quadric in Five-Axis Machining," Comput.-Aided Des., **53**, pp. 126–138.

[19] Lu, Y., Ding, Y., and Zhu, L., 2016, "Simultaneous Optimization of the Feed Direction and Tool Orientation in Five-Axis Flat-end Milling," Int. J. Prod. Res., **54**(15), pp. 4537–4546.

[20] Lu, Y. A., Ding, Y., and Zhu, L. M., 2017, "Tool Path Generation via the Multi-Criteria Optimisation for Flat-end Milling of Sculptured Surfaces," Int. J. Prod. Res., **55**(15), pp. 4261–4282.

[21] Liang, F., Kang, C., and Fang, F., 2021, "A Review on Tool Orientation Planning in Multi-Axis Machining," Int. J. Prod. Res., **59**(18), pp. 5690–5720.

[22] Warkentin, A., Ismail, F., and Bedi, S., 2000, "Comparison Between Multi-Point and Other 5-Axis Tool Positioning Strategies," Int. J. Mach. Tools Manuf., **40**(2), pp. 185–208.

[23] Duvedi, R. K., Bedi, S., and Mann, S., 2018, "An Efficient Multipoint 5-Axis Tool Positioning Method for Tensor Product Surfaces," Int. J. Adv. Manuf. Technol., **97**(1–4), pp. 279–295.

[24] Gray, P. J., Ismail, F., and Bedi, S., 2004, "Graphics-Assisted Rolling Ball Method for 5-Axis Surface Machining," Comput.-Aided Des., **36**(7), pp. 653–663.

[25] Gray, P., Bedi, S., and Ismail, F., 2003, "Rolling Ball Method for 5-Axis Surface Machining," Comput.-Aided Des., **35**(4), pp. 347–357.

[26] Gray, P. J., Bedi, S., and Ismail, F., 2005, "Arc-Intersect Method for 5-Axis Tool Positioning," Comput.-Aided Des., **37**(7), pp. 663–674.

[27] Li, H., and Feng, H. Y., 2004, "Efficient Five-Axis Machining of Free-Form Surfaces With Constant Scallop Height Tool Paths," Int. J. Prod. Res., **42**(12), pp. 2403–2417.

[28] Hosseinkhani, Y., Akbari, J., and Vafaeesefat, A., 2007, "Penetration-Elimination Method for Five-Axis CNC Machining of Sculptured Surfaces," Int. J. Mach. Tools Manuf., **47**(10), pp. 1625–1635.

[29] Yoon, J. H., Pottmann, H., and Lee, Y. S., 2003, "Locally Optimal Cutting Positions for 5-Axis Sculptured Surface Machining," Comput.-Aided Des., **35**(1), pp. 69–81.

[30] Chiou, C. J., and Lee, Y. S., 1999, "A Shape-Generating Approach for Multi-Axis Machining G-Buffer Models," Comput.-Aided Des., **31**(12), pp. 761–776.

[31] Gautschi, W., 2011, *Numerical Analysis*, Springer Science & Business Media, New York.

[32] Lynch, K. M., and Park, F. C., 2017, *Modern Robotics*, Cambridge University Press, Cambridge.

[33] Pottmann, H., and Hofer, M., 2003, "Geometry of the Squared Distance Function to Curves and Surfaces," *Visualization and Mathematics III*, Springer, New York, pp. 221–242.

[34] Chen, Y., Huang, P., and Ding, Y., 2022, "An Analytical Method for Corner Smoothing of Five-Axis Linear Paths Using the Conformal Geometric Algebra," Comput.-Aided Des., **153**, p. 103408.

[35] Kleppe, A. L., Tingelstad, L., and Egeland, O., 2019, "Coarse Alignment for Model Fitting of Point Clouds Using a Curvature-Based Descriptor," IEEE Trans. Autom. Sci. Eng., **16**(2), pp. 811–824.

[36] Kanatani, K., 2015, *Understanding Geometric Algebra: Hamilton, Grassmann, and Clifford for Computer Vision and Graphics*, CRC Press, Boca Raton, FL.